Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Please cancel claims 8, 50, 52, 58, and 65-67, amend claims 6, 56, and 69, and add new claims 74-80 as follows:

Claims 1-5 (canceled)

6. (original): A method of supporting conditional execution in a very long instruction word (VLIW) based array processor~~with subword execution~~, the method comprising:

executing a first instruction that identifies an execution unit from a plurality of execution units as the identified execution unit affecting the value of an arithmetic condition flag (ACF), wherein the identified execution unit is associated with a second instruction in a VLIW;

executing the second instruction by the identified execution unit, the second instruction identifying a condition resulting from the execution of the second instruction; and

~~providing general purpose flag bits (ACFs) that contain reduced condition information that is used for branching or conditional execution; and~~

~~specifying and~~ setting ~~a~~ the identified condition in the ~~ACFs~~ ACF ~~based upon a condition code specification encoded in an instruction generating a condition~~.

7. (original): The method of claim 6 wherein instructions that execute conditionally do not affect the ACFs.

8. (canceled)

9. (original): The method of claim 6 further comprising the steps of:

executing a packed data instruction where the execution of each sub-word of the packed data operation is dependent upon the associated subword ACF.

Claims 10-48 (canceled)

49. (original): An indirect very long instruction word (VLIW) processing system comprising:

a first processing element (PE) having a VLIW instruction memory (VIM) for storing instructions in slots within a VIM memory locations;

a first register for storing a function instruction having a plurality of group bits defining instruction type and a plurality of unit field bits defining execution unit type;

a predecoder for decoding the plurality of group bits and the plurality of unit field bits; and

a load mechanism for loading the function instruction in an appropriate one of said slots in VIM based upon said decoding, the first processor further comprising:

at least two execution units, each execution unit receiving at least two operands from a register file;

each execution unit having instruction control lines derived from a registered instruction in a processor pipeline, the instruction control lines including conditional execution control lines to control conditional operation as specified in an instruction to be executed;

each execution unit producing a result and a latched arithmetic scalar condition state;

each execution unit having a first latch for holding the arithmetic scalar condition state for the instruction after the instruction has finished its execution state;

each execution unit having a second latch connected to the conditional execution control lines for holding instruction control signals for the instruction after the instruction has finished its execution state;

each execution unit having an arithmetic condition flag (ACF) generation unit for providing a Boolean combination of a present selected state with a previous state; and

a single ACF latch for all of the execution units for storing the previous state and feeding the previous state back to the respective ACF generation unit.

50. (canceled)

51. (original): The system of claim 49 wherein the PE further comprises a multiplexer connected to receive said Boolean combination from each of the ACF generation units and to controllably switch said Boolean combinations to branch logic in a sequence processor (SP).

52. (canceled)

Claims 53-55 (canceled)

56. (currently amended): An apparatus for supporting conditional execution in a very long instruction word (VLIW) based array processor~~ with subword execution~~, the apparatus comprising:

a processing element (PE) for providing ~~general purpose flag bits~~ arithmetic condition flags (ACFs), the PE having a plurality of execution units that produce condition as a result of the execution units executing; ~~that contain reduced condition information that is used for branching or conditional execution; and~~

a first instruction identifying one of the execution units as the identified execution unit to affect the value of an ACF; and

~~an~~ a second instruction having a condition code specification encoded therein identifying a condition from the identified execution unit, the processing element ~~specifying and~~ setting ~~a~~ the identified condition in ACFs ~~based upon the condition code specification generating a condition~~ upon execution of the second instruction.

57. (previously presented): The apparatus of claim 56 wherein instructions that execute conditionally do not affect the ACFs.

58. (canceled)

59. (previously presented): The apparatus of claim 56 further comprising the steps of:

executing a packed data instruction where the execution of each sub-word of the packed data operation is dependent upon the associated subword ACF.

60. (previously presented): An indirect very long instruction word (VLIW) processing system comprising:

a fetch controller;

a VLIW instruction memory (VIM);

a processing element (PE) having a VIM controller, and a plurality of execution units, the VIM controller receiving a VLIW instruction from the fetch controller and generating VIM addresses for segmenting the VLIW instruction and storing the segmented VLIW instruction into slots within a VIM memory location, each slot corresponding to each execution unit;

each execution unit having a plurality of instruction control lines through which to load an instruction stored in the execution unit's corresponding slot, a portion of said plurality of instruction control lines carrying an instruction control signal for controlling conditional operation as specified in the loaded instruction, each execution unit receiving as input at least two operands and an arithmetic condition flag (ACF), the ACF representing a previous state of the processing element, each execution unit producing a result defining an execution state; and

an ACF latch connected to each execution unit, the ACF latch storing the previous state and feeding the previous state back to each execution unit.

61. (previously presented): The indirect VLIW processing system of claim 60 wherein each execution unit producing an arithmetic scalar condition state.

62. (previously presented): The indirect VLIW processing system of claim 61 wherein each execution unit further comprises a first latch storing the arithmetic scalar condition state.

63. (previously presented): The indirect VLIW processing system of claim 62 wherein at least one of said plurality of execution units further comprises a second latch connected to the portion of said plurality of instruction control lines for holding the instruction control signal for the instruction after the instruction has finished its execution state.

64. (previously presented): The indirect VLIW processing system of claim 63 wherein the at least one of said plurality of execution units further comprises an ACF generation unit for providing a Boolean combination of a present selected state with a previous state.

Claims 65-67 (canceled)

68. (previously presented): The indirect VLIW processing system of claim 64 wherein the PE further comprises a multiplexer connected to receive said Boolean combination from each of the ACF generation units and to controllably switch said Boolean combinations to branch logic in a sequence processor (SP).

69. (currently amended): A method of supporting conditional execution in a very long instruction word (VLIW) based array processor, the VLIW based array processor having a processor element, the method comprising:

receiving a VLIW instruction;

generating VIM addresses for segmenting the VLIW instruction and storing the segmented VLIW instruction into slots within a VIM memory location;

loading an instruction stored in one of the slots;

receiving an instruction control signal for controlling conditional operation as specified in the loaded instruction;

receiving as input at least two operands and an arithmetic condition flag (ACF), the ACF representing a previous condition state of the processing element;

producing a result based on the loaded instruction, said at least two operands and the ACF, the result defining an execution condition state of the instruction; and

storing the condition state in the ACF to be used as the previous condition state of the processing element for a subsequent producing step.

70. (previously presented): The method of claim 69 wherein the producing step further comprises producing an arithmetic scalar condition state.

71. (previously presented): The method of 70 wherein the storing step further comprises storing the arithmetic scalar condition state.

72. (previously presented): The method of claim 71 wherein the storing step a latch is used for storing the instruction control signal for the instruction after the instruction has entered the execution state.

73. (previously presented): The method of claim 72 wherein the producing step further comprises providing a Boolean combination of a present selected state with a previous state.

74. (new): The method of claim 6 further comprising:

executing a third instruction following the execution of the second instruction, the third instruction based on the ACF having a bit field that selects the value of the ACF to determine whether to execute the third instruction or not.

75. (new): The method of claim 6 wherein the first instruction is an execute VLIW (XV) instruction.

76. (new): The method of claim 6 wherein the first instruction is a load VLIW (LV) instruction.

77. (new): The method of claim 6 wherein the execution unit affecting the value of an arithmetic condition flag (ACF) is identified by a unit affecting field (UAF) bit field in the first instruction.

78. (new): The method of claim 6 wherein the second instruction is a compare instruction.

79. (new): The method of claim 6 wherein the second instruction is an arithmetic logic unit (ALU), multiply accumulate unit (MAU), or data select unit (DSU) instruction.

80. (new): The method of claim 74 wherein the third instruction is an arithmetic logic unit (ALU), multiply accumulate unit (MAU), data select unit (DSU), a load, or a store instruction.